ECOLE POLYTECHNIQUE DE TUNISIE

Denoising Dental X-ray Images

Atef BOUZID, Rayhan BEN YOUSSEF

Supervised by: Ms. Ines BOUSNINA

2023-2024

"Medical imaging techniques, including X-rays, Magnetic Resonance Imaging (MRI), Computer Tomography (CT), and ultrasound, are inherently susceptible to noise. This can stem from various factors, such as the use of different image acquisition techniques or efforts to minimize patient exposure to radiation. However, reducing radiation often comes at the cost of increased noise. This noise can hinder accurate image analysis, both by human experts and automated systems. Denoising techniques are therefore crucial for enhancing the quality and interpretability of medical images, ensuring their reliable use in clinical diagnosis and research."

1 Introduction

Image denoising, a longstanding problem in computer vision, has been tackled using diverse techniques. These include models based on partial differential equations (PDEs), domain transformations (e.g., wavelets, DCT), non-local methods (e.g., NL-means), hybrid approaches (e.g., BM3D), and sparse coding-based methods. Despite their differences, they share a common goal: to restore images by removing noise while preserving essential features.

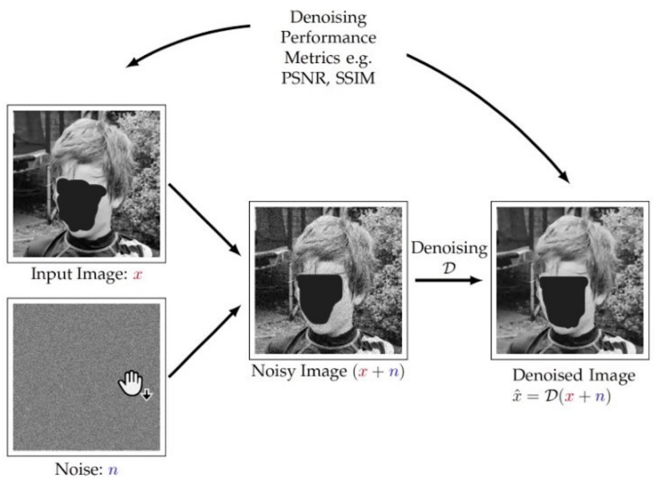

Figure 1: Image Denoising: The Basic Idea

The image denoising problem can be mathematically formulated as:

$$y = x + n \tag{1}$$

where y represents the noisy image, x denotes the original image, and n signifies the noise. Most denoising methods aim to approximate x as closely as possible using the observed y. In most cases, the noise n is assumed to be generated from a well-defined process.

This paper investigates the denoising performance of deep learning autoencoders compared to established methods like BM3D on dental X-rays. We utilize the Dental X-ray dataset, deliberately introducing various noise types, and then compare the effectiveness of each approach in reducing noise.

2 Image Noise

Random variations in image brightness, known as noise [1], are typically caused by the sensors and circuits inside scanners and cameras. To analyze and evaluate denoising performance, we have incorporated multiple noise types into our dataset, as detailed below.

2.1 Additive White Gaussian Noise (AWGN)

AWGN models each pixel (m, n) of the observation y by the sum of the pixel (m, n) of the noiseless image x and of a pixel of the noise b:

$$y(m, n) = x(m, n) + b(m, n) \quad \forall m, n \tag{2}$$

where $b(m, n) \sim \mathcal{N}(0, \sigma^2)$.

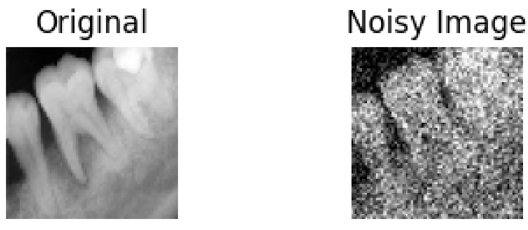

Figure 2: Image with Additive Gaussian Noise ($\mu = 0$, $\sigma = 0.2$)

2.2 Poisson noise (also called shot noise)

Poisson noise models the acquisition of photons on a photosite. The number of photons is random and depends on the illumination. The corresponding Poisson process has a mean equals to the illumination. The intensity of each pixel (m, n) of the observation y is:

$$y(m, n) \sim \mathcal{P}(x(m, n)).$$

This model is used in the case of acquisitions with a low number of photons.

- **Note :**

As a reminder, the Poisson distribution $\mathcal{P}(X)$ writes

$$p(X = k) = \frac{e^{-\lambda}\lambda^k}{k!}.$$

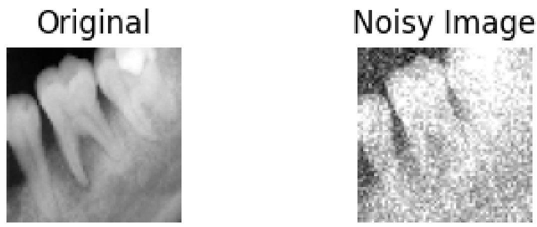

Figure 3: Image with Normalized Poisson Noise ($\lambda = 5$)

3 Selected Methods

3.1 BM3D

BM3D is a recent nonlocal image modeling technique based on the fact that an image has a locally sparse representation in transform domain [2]. This algorithm can be split into two major steps. The first step estimates the de-noised image using hard thresholding during the collaborative filtering and can be divided into three minor steps. Starting by finding the image patches similar to a given image patch and grouping them in a 3D block, then a 3D linear transform (Wavelet transform) and shrinkage of coefficient are applied to 3D block . The final 3D block is given by filtering out simultaneously all 2D image patches. The second major step is carried out by using Wiener filtering. This second step imitates the first one with two differences [3]. In fact the filtered patches are compared instead of the original patches. Moreover the new 3D group is processed by Wiener filtering instead of a mere threshold. The algorithm is summarized in the figure 4

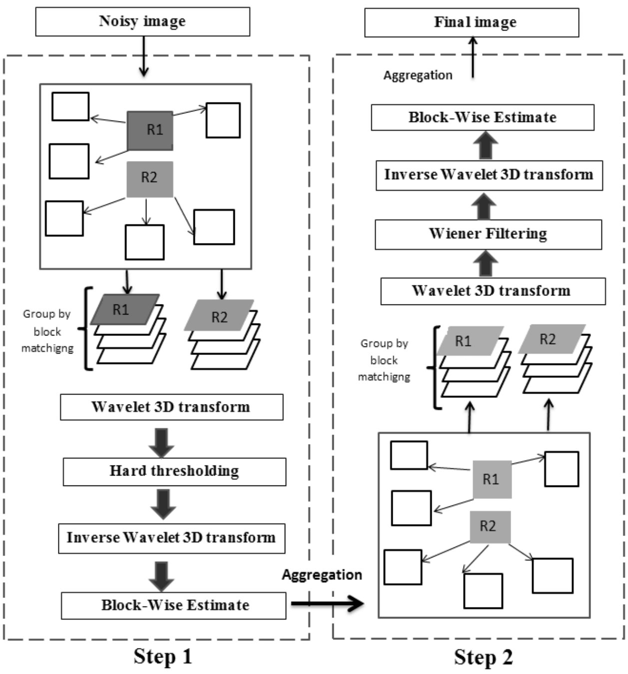

Figure 4: Diagram of the BM3D algorithm.

3.2 Auto-Encoders

An autoencoder is a type of neural network that tries to learn an approximation to the identity function using backpropagation [4]. Given a set of unlabeled training inputs $x^{(1)}, x^{(2)}, \ldots, x^{(n)}$, it aims to reconstruct each input at the output layer, i.e.,

$$z^{(i)} \approx x^{(i)} \quad \text{for } i = 1, 2, \ldots, n. \tag{3}$$

An autoencoder consists of two main parts:

1. **Encoder:** Maps the input $x \in [0, 1]^d$ to a hidden representation $y \in [0, 1]^{d_0}$ using a deterministic mapping, often a non-linear function:

$$y = s(Wx + b), \tag{4}$$

where W is the weight matrix, b is the bias vector, and s is the activation function.

2. **Decoder:** Reconstructs the input from the latent representation y, producing an output z of the same shape as x:

$$z = s(W'y + b'). \tag{5}$$

Note that the prime symbol in W' does not denote matrix transpose.

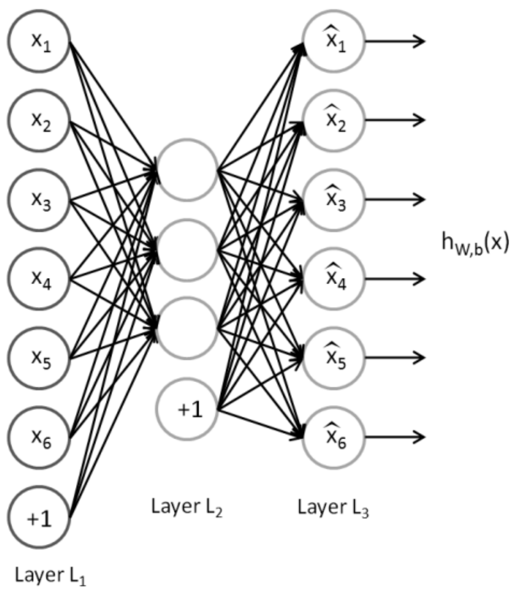

Figure 5: Basic architecture of an autoencoder.

Key Points:

- Layer L1 is the input layer, where data is fed into the model.
- Layer L2 is the hidden layer, representing the latent representation of the input.
- Layer L3 is the output layer, where the reconstructed input is produced.
- Using fewer hidden units than input units forces the autoencoder to learn a compressed representation, similar to Principal Component Analysis (PCA).
- Having more hidden units than input units can still be useful for discovering insights if certain sparsity constraints are imposed.

3.3 Variants of Autoencoders

3.3.1 Denoising Autoencoders

Denoising autoencoders are a stochastic extension to classic autoencoders, aiming to reconstruct the input from a corrupted version. This is achieved by:

1. Randomly setting some of the inputs to zero, simulating missing or corrupted data.
2. Training the autoencoder to predict the original, uncorrupted values for these missing inputs.

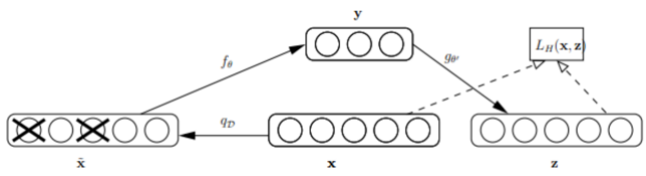

Figure 6: Basic architecture of a denoising autoencoder.

Denoising autoencoders can be stacked to create deep networks, known as stacked denoising autoencoders.

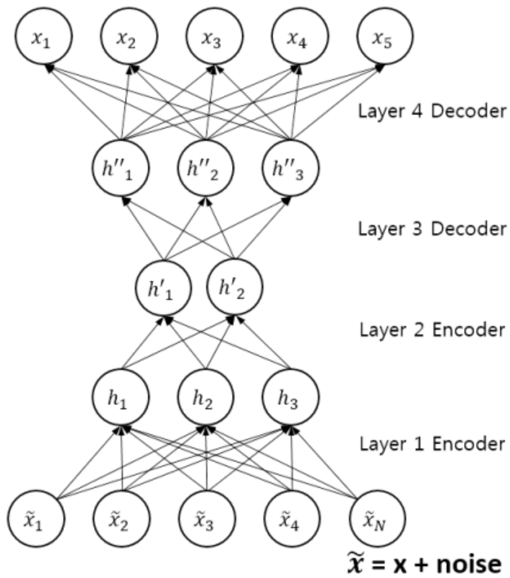

Figure 7: A stacked denoising autoencoder.

3.3.2 Convolutional Autoencoders

Convolutional autoencoders (CAEs) are particularly well-suited for image processing tasks as they leverage the advantages of convolutional neural networks (CNNs)[5]. Key features of CAEs include:

- Convolutional encoding and decoding layers, preserving spatial relationships in images.
- Weight sharing across input locations, promoting local spatiality.

The representation of the ith feature map in a CAE is given by:

$$h_i = s(x * W_i + b_i), \tag{6}$$

where $*$ denotes convolution (2D), s is an activation function, W_i is the weight matrix for the ith filter, and b_i is the bias for the ith filter.

The reconstruction of the input is obtained as:

$$y = s(\sum_{i \in H} h_i * \tilde{W}_i + c), \tag{7}$$

where H is the group of latent feature maps, $\tilde{W}_i$ is the flipped version of W_i, and c is the bias per input channel.

Backpropagation is used to compute the gradients of the error function with respect to the model parameters.

4 Performance Measurement Metrics

The performance of our experiments are measured using both subjective and objective fidelity criteria. Objective fidelity criteria gives us blind results, which is good for understanding the differences between the outputs and for subjective fidelity criteria, we observed how human eyes perceive the denoising performance.

4.1 Objective Fidelity Criteria

When it comes to measuring the success of image denoising, two objective metrics steal the show: PSNR and SSIM [6].

- **Peak Signal to Noise Ratio (PSNR)**

The peak signal to noise ratio (PSNR) represents the ratio between the maximum power of a signal to the noise which degrades the original image. This measure is based on the Mean Squared Error (MSE) which assesses the difference between the original image data and the degraded image data. In this measure, a higher value indicates a better denoised image.
PSNR of a denoised image is calculated as:

$$\text{PSNR} = 10 \log_{10} \frac{\text{MAX}^2}{\text{MSE}}, \tag{8}$$

where MAX indicates the maximum intensity of the image. For a standard grayscale image, it is 255.

If $y(x)$ is an original image of size $M \times N$ and $\hat{y}(x)$ is the denoised image, then MSE is calculated as:

$$\text{MSE} = \frac{1}{M \times N} \sum_{x \in X} (y(x) - \hat{y}(x))^2. \tag{9}$$

- **Structural Similarity (SSIM) Index**

Although PSNR is a good quality measure, it can sometimes be misleading because it considers only intensity values and not structural similarity between images. To address this, Wang et al proposed the Structural Similarity (SSIM) index, which incorporates geometric and structural comparisons. SSIM is defined as:

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{10}$$

where:

- μ_x is the pixel sample mean of x
- μ_y is the pixel sample mean of y
- σ_x^2 is the variance of x

- σ_y^2 is the variance of y
- σ_{xy} is the covariance of x and y
- $c_1 = (k_1 L)^2$, $c_2 = (k_2 L)^2$ are variables to stabilize the division with weak denominator
- L is the dynamic range of the pixel-values (typically $2^{\#bits\ per\ pixel} - 1$)
- $k_1 = 0.01$ and $k_2 = 0.03$ by default

4.2 Subjective Fidelity Criteria

To fully judge how well our denoising techniques worked, we didn't just rely on numbers. We also looked at the cleaned-up images with our own eyes, comparing them side-by-side and zooming in on key areas. This helped us see how much noise each method removed, whether it kept important details clear, and how natural the final image looked overall.

5 BM3D: Experimental Setup, Results, and Analysis

5.1 Experimental Setup

- *Implementation:* The BM3D algorithm was implemented using the `bm3d` library in Python.
- *Input Images:* A set of dental X-ray images containing various types of noise, such as Gaussian and Poisson noise.
- *Denoising Function:* `bm3d.bm3d(noisy_images[i], sigma_psd=0.2, stage_arg=bm3d.BM3DStages.HARD_THRESHOLDING)`
- *Parameters:*
  - `sigma_psd`: Set to 0.2, indicating an estimated standard deviation of noise of 0.2.
  - `stage_arg`: Set to `bm3d.BM3DStages.HARD_THRESHOLDING` for aggressive noise suppression in the first stage.

5.2 Results

We selected a single image from the dataset (Figure 8).

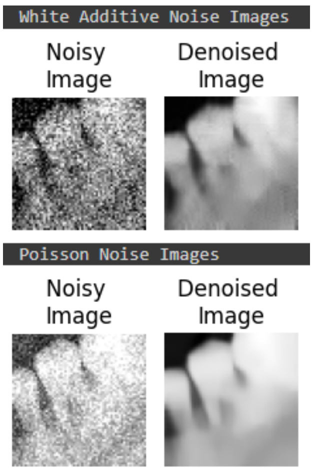

Figure 8: BM3D Results

5.3 Quantitative Metrics

Key Observations:

- BM3D significantly improved PSNR and SSIM (Tables 1 and 2).
- Visual comparisons for PSNR and SSIM are shown in Figures 9 and 10.

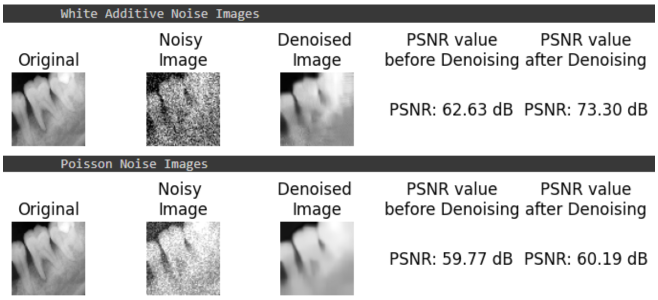

Figure 9: PSNR Comparison: Original, Noisy, and Denoised Images

Table 1: PSNR Improvement with BM3D Denoising

Noise type	PSNR	Increase (%)
AWGN	73.30 dB	17.04
Poisson	60.19 dB	0.7

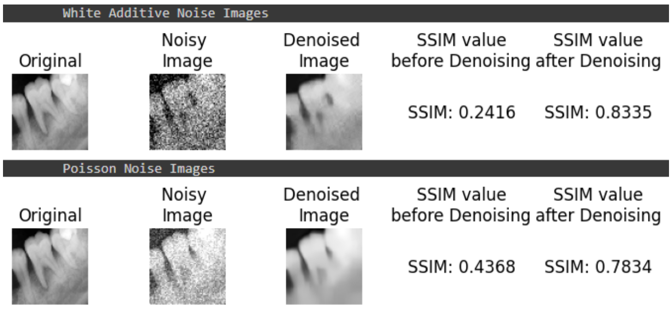

Figure 10: SSIM Comparison: Original, Noisy, and Denoised Images

Table 2: SSIM Improvement with BM3D Denoising

Noise type	SSIM	Increase (%)
AWGN	0.8335	245
Poisson	0.7834	79.56

5.3.1 Qualitative Analysis

As we check out the original, noisy, and denoised versions using the BM3D method, in the denoised one, we notice things getting way clearer. It brings out details that were kinda hidden.

We expected the denoised version to make things sharper, and it does. It highlights finer details within these versions.

But, here's the thing – in some spots with strong differences, like edges and shadows, we notice these little quirks. It's like the denoising process leaves a tiny mark there.

6 Autoencoder: Experimental Setup, Results, and Analysis

6.1 Experimental Setup

- *Implementation:* The autoencoder was implemented using TensorFlow and Keras in Python.
- *Number of parameters:* 74497
- *Dataset* 120 748x512 dental X-ray images containing white additive Gaussian noise.
- *Architecture:* The autoencoder employs a convolutional architecture with the following layers:
  - Encoder:
    - Conv2D (32 filters, 3x3 kernel, ReLU activation)
    - MaxPooling2D (2x2 pool size)
    - Conv2D (64 filters, 3x3 kernel, ReLU activation)
    - MaxPooling2D (2x2 pool size)
  - Decoder:
    - Conv2D (64 filters, 3x3 kernel, ReLU activation)
    - UpSampling2D (2x2 upsampling)
    - Conv2D (32 filters, 3x3 kernel, ReLU activation)
    - UpSampling2D (2x2 upsampling)
    - Conv2D (1 filter, 3x3 kernel, sigmoid activation)
- *Training Parameters:*
  - Optimizer: Adam with a learning rate of 0.001
  - Loss function: Mean squared error
  - Metrics: Mean absolute error
  - Epochs: 40
  - Batch size: 128

6.2 Results

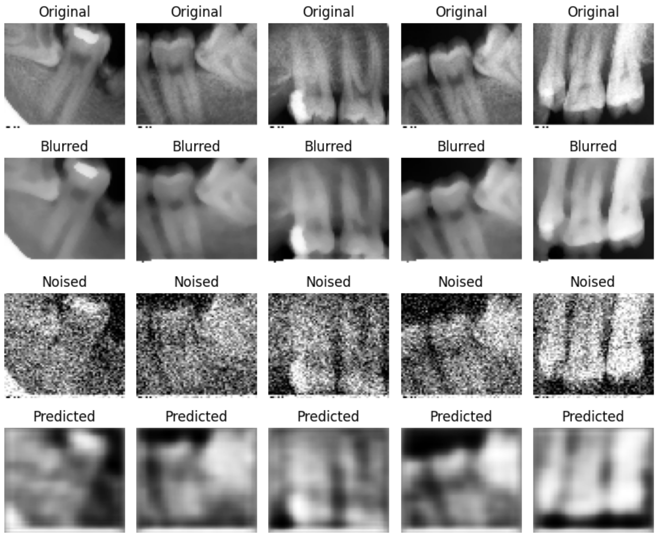

Figure 11: Original vs. Denoised (autoencoder)

6.2.1 Quantitative Metrics

Table 3: Comparison of Mean PSNR Scores

Image type	Mean PSNR score
Noisy	57.06 dB
Blurry (Median Blur)	62.78 dB
Autoencoder	68.43 dB

Table 4: Comparison of Mean SSIM Scores

Image type	Mean SSIM score
Noisy	0.2752
Autoencoder	0.7256

6.2.2 Qualitative Analysis

In looking at the noisy, original, and autoencoder (denoised/predicted) images, we noticed something unexpected. Despite thinking the autoencoder would make things clearer, the denoised image turned out to be a bit blurry and not very clear. It's so unclear that it's tough to figure out if it's a dental X-ray. We're thinking this might be because our dataset is on the smaller side.

So, we tried out a technique called medianBlur, and you can see the results in Figure 11. Upon close inspection, the blurred image is a bit fuzzy too, but surprisingly, it's clearer than the autoencoder output. Although the PSNR value indicates that the blurred image isn't as good as the autoencoder,(refer to Table 4 for details), our eyes are telling a different story. It's like a gap between what we see and what the numbers are saying, making us dig deeper into what's influencing our results.

7 Simulation results

As we saw in the previous sections, when visually comparing the results, the BM3D method clearly outperforms the autoencoder in our image denoising task. However, as we mentioned earlier, relying solely on mathematical outcomes may not always be sufficient to make the ultimate decision.

The BM3D method doesn't require a pre-existing dataset and works on any type of image. On the other hand, after conducting research to understand why the autoencoder wasn't delivering good results, we found that the autoencoder method requires a sufficiently large "training set" to train the model effectively, not a small dataset like in our case. We chose to challenge ourselves by opting for a small dataset initially, curious about the outcomes. However, after seeing the results, we are now convinced of the importance of having a larger dataset for better outcomes.

Additionally, the BM3D method doesn't require powerful computers to execute and provides nearly instant results, unlike the autoencoder method. The autoencoder demands significant resources during the training phase, particularly with a large dataset. However, in our case, since we have a small dataset, the autoencoder didn't take much time. We focused on preventing overfitting, which is why we selected the epoch number as shown in Figure 12. In that figure, you can observe the Mean Absolute Error (MAE) function in relation to the number of epochs.

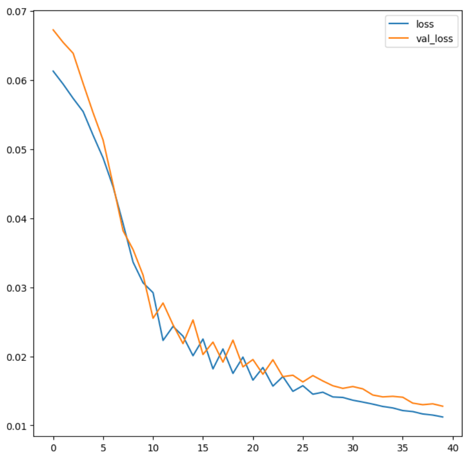

Figure 12: Training and Validation Loss for Autoencoder

8 Conclusion

This project provided us with a valuable opportunity to delve into the practical aspects of the Signal Processing field, particularly in Image Processing. Fueled by curiosity, we explored the realm of Image Denoising during the research phase of the project, uncovering a spectrum of methods used over the decades – from traditional filters to cutting-edge non-local means techniques like Block-Matching and 3D Filtering, as well as emerging Deep Learning algorithms.

In the course of this project, we implemented and trained a Convolutional Auto-Encoder for Image Denoising. This implementation was then compared to another technique, and its robustness to various noise levels was meticulously measured.

While our work represents a significant step in our journey, it is just the beginning of what we aspire to achieve. Due to the limited availability of medical datasets, our choices were constrained. Our future goal is to propose and explore more in the medical domain, and perhaps even integrate aspects into the healthcare sector.

References

[1] Basics of Image Processing. [Online]. Available: https://vincmazet.github.io/bip/restoration/denoising.html (Accessed on December 18, 2023)

[2] K. D. and al. (2008) Image denoising by sparse 3-d transform-domain collaborative filtering. [Online]. Available: Image Denoising by Sparse 3-D Transform-Domain Collaborative Filtering (Accessed on December 18, 2023)

[3] Marc Lebrun, 'An Analysis and Implementation of the BM3D Image Denoising Method', Image Processing On Line, 2 (2012), pp. 175–213. Fengqing Qin. "Blind Image Restoration Based on Wiener Filtering and Defocus Point Spread Function Estimation." 5th International Congress on Image and Signal Processing (CISP 2012). pp. 360–363. Chongqing, December 25, 2023.

[4] Deep learning tutorial, Stanford University. Autoencoders. Available: http://ufldl.stanford.edu/tutorial/unsupervised/Autoencoders (Accessed on December 26, 2023)

[5] A. Ng. (2017) Coursera: Neural networks and deep learning. [Online]. Available: https://www.coursera.org/learn/neural-networks-deep-learning (Accessed on December 26, 2023)

[6] A. H. and D. Z. (2010) Image quality metrics: PSNR vs. SSIM [Online]. Available: https://www.researchgate.net/publication/220931731_Image_quality_metrics_PSNR_vs_SSIM (Accessed on December 28, 2023)